Guidelines

Description

Provides information and data to educate software development professionals on the concept, applicability and value of design guidelines. Furthermore, this section collects and makes available a set of Design Guidelines to assist software development professionals (architects, designers, developers, QA, auditors, etc.) in identifying and removing potential classes of vulnerabilities in the software systems they are building, as well as developing more mature and security-knowledge-aware design practices for future software systems.

Overview Articles

Name	Version Creation Time	Abstract
Guidelines Overview	11/14/08 5:03:42 PM	All systems have vulnerabilities, either in the technology from which they are constructed or in the behaviors of the people who use them.

Most Recently Updated Articles [Ordered by Last Modified Date]

Name	Version Creation Time	Abstract
Use Well-Known Cryptography Appropriately and Correctly	11/14/08 5:07:34 PM	Failing to use, or inventing your own, cryptography can introduce vulnerability.
Use Authorization Mechanisms Correctly	11/14/08 5:07:06 PM	Incorrect use of, or failing to use, authorization mechanisms can introduce vulnerability.
Use Authentication Mechanisms, Where Appropriate, Correctly	11/14/08 5:06:36 PM	Incorrectly using, or failing to use, authentication mechanisms can introduce vulnerability.
Treat the Entire Inherited Process Context as Unvalidated Input	11/14/08 5:06:10 PM	Inherited process context that is not validated like other inputs can introduce vulnerability.
Never Use Unvalidated Input as Part of a Directive to any Internal Component	11/14/08 5:05:43 PM	Using unvalidated input as part of a directive or command to a subsystem can introduce vulnerability.

All Articles [Ordered by Title]

Name	Version Creation Time	Abstract
Assume that Human Behavior Will Introduce Vulnerabilities into Your System	11/14/08 4:58:50 PM	People introduce vulnerability.

Be Suspicious about Trusting Unauthenticated External Representation of Internal Data Structures	11/14/08 4:59:22 PM	Trusting unauthenticated externalized data structures can introduce vulnerability.
Carefully Study Other Systems Before Incorporating Them into Your System	11/14/08 4:59:54 PM	Indiscriminate delegation of function to other systems can introduce vulnerabilities.
Clear Discarded Storage that Contained Secrets and Do Not Read Uninitialized Storage	11/14/08 5:00:21 PM	Failing to initialize storage can introduce vulnerability.
Design Configuration Subsystems Correctly and Distribute Safe Default Configurations	11/14/08 5:00:50 PM	Poorly designed configuration subsystems and poor default configurations may produce system vulnerabilities.
Do Not Perform Arithmetic with Unvalidated Input	11/14/08 5:01:20 PM	Careless modulo arithmetic can introduce vulnerability.
Do Not Use the "%n" Format String Specifier	11/14/08 5:01:49 PM	Careless use of "%n" format strings can introduce vulnerability.
Ensure that Input Is Properly Canonicalized	11/14/08 5:02:19 PM	Failure to canonicalize input can introduce vulnerability. Inadvertently canonicalizing input multiple times can introduce vulnerability.
Ensure that the Bounds of No Memory Region Are Violated	11/14/08 5:02:47 PM	Violation of memory bounds can introduce vulnerability.
Follow the Rules Regarding Concurrency Management	11/14/08 5:03:15 PM	Failure to follow proper concurrency management protocols can produce serious vulnerabilities. Concurrent access to shared resources without using appropriate concurrency management mechanisms produces hard-to-find vulnerabilities. Many "functions" that are necessary to use can introduce "time of check/time of use" vulnerabilities.
Guidelines Overview	11/14/08 5:03:42 PM	All systems have vulnerabilities, either in the technology from which they are constructed or in the behaviors of the people who use them.
Handle All Errors Safely	11/14/08 5:04:45 PM	Unhandled or incorrectly handled exceptions can introduce vulnerability.
If Emulation of Another System Is Necessary, Ensure that It Is as Correct and Complete as Possible	11/14/08 5:05:15 PM	Incorrect or incomplete emulation can introduce vulnerability.

Never Use Unvalidated Input as Part of a Directive to any Internal Component	11/14/08 5:05:43 PM	Using unvalidated input as part of a directive or command to a subsystem can introduce vulnerability.
Treat the Entire Inherited Process Context as Unvalidated Input	11/14/08 5:06:10 PM	Inherited process context that is not validated like other inputs can introduce vulnerability.
Use Authentication Mechanisms, Where Appropriate, Correctly	11/14/08 5:06:36 PM	Incorrectly using, or failing to use, authentication mechanisms can introduce vulnerability.
Use Authorization Mechanisms Correctly	11/14/08 5:07:06 PM	Incorrect use of, or failing to use, authorization mechanisms can introduce vulnerability.
Use Well-Known Cryptography Appropriately and Correctly	11/14/08 5:07:34 PM	Failing to use, or inventing your own, cryptography can introduce vulnerability.